

Documentation Technique

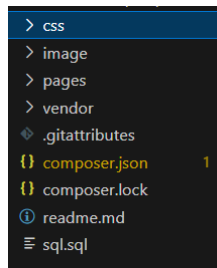
Nom (du projet) : HB Commerce

Introduction :

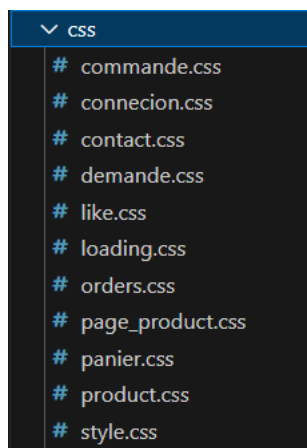
Ce document fournit une documentation technique complète de l'architecture, du code et des configurations du projet e-commerce nommé "HB Commerce". Le projet est structuré en plusieurs dossiers et fichiers, comprenant des scripts PHP pour les fonctionnalités du site, des fichiers CSS pour le style, et des configurations nécessaires pour le bon fonctionnement du projet.

Structure du Projet :

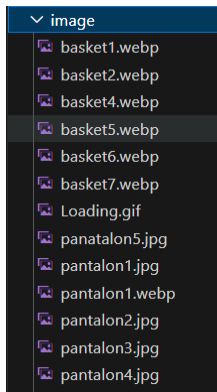
Le projet "**HB Commerce**" est organisé en plusieurs dossiers et fichiers pour maintenir une structure claire et fonctionnelle. À la racine du projet, on trouve deux fichiers essentiels : `sql.sql` et `readme.md`. Le fichier `sql.sql` contient les scripts SQL nécessaires pour créer et configurer la base de données du projet. Le fichier `readme.md` fournit une documentation de base pour l'installation et la configuration du projet, incluant des instructions spécifiques pour la configuration de XAMPP, l'utilisation du système de paiement PayPal sandbox, et une description des fonctionnalités optionnelles du site.



Le dossier `css/` contient tous les fichiers CSS utilisés pour le style des pages web. Chaque fichier CSS est dédié à une section ou fonctionnalité spécifique du site, ce qui permet de maintenir un style cohérent et organisé. Les fichiers CSS incluent `commande.css`, `connexion.css`, `contact.css`, `demande.css`, `like.css`, `loading.css`, `order.css`, `page_product.css`, `panier.css`, `product.css` et `style.css`.



Le dossier `image/` contient toutes les images utilisées sur le site web, telles que les images des produits, les icônes et les bannières. Ces ressources graphiques sont essentielles pour offrir une expérience utilisateur visuellement attrayante et professionnelle.

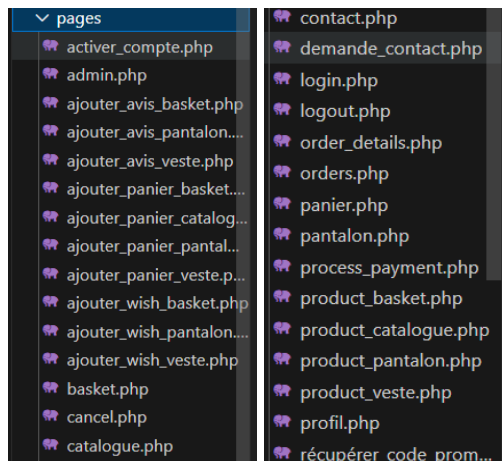


Le dossier `pages/` est le plus volumineux et contient tous les fichiers PHP qui implémentent les fonctionnalités du site. Chaque fichier PHP dans ce dossier gère une fonctionnalité ou une page spécifique du site web. Par exemple, `activer_compte.php` gère l'activation des comptes utilisateurs via un lien d'activation envoyé par email, tandis que `admin.php` fournit une interface d'administration pour gérer les utilisateurs, les produits et les commandes.

D'autres fichiers PHP importants incluent `ajouter_avis_basket.php`, `ajouter_avis_pantalon.php`, `ajouter_avis_veste.php` qui permettent aux utilisateurs d'ajouter des avis sur les produits, et `ajouter_panier_basket.php`, `ajouter_panier_veste.php`, `ajouter_panier_pantalon.php` qui gèrent l'ajout de produits au panier des utilisateurs. Des fichiers comme `catalogue.php` affichent le catalogue de produits sur la page d'accueil, tandis que `commande.php`, `order.php` et `order_detail.php` gèrent les commandes et les paiements des utilisateurs.

Les fichiers `login.php` et `logout.php` gèrent l'authentification des utilisateurs, permettant la connexion et la déconnexion. Les fichiers `panier.php`, `wish.php`, `profil.php` affichent respectivement le panier, la liste de souhaits et le profil de l'utilisateur, incluant des fonctionnalités comme l'application de codes promo et l'affichage des points de fidélité.

En résumé, la structure du projet est conçue pour être modulaire et organisée, facilitant ainsi la gestion, la maintenance et l'évolution du site e-commerce.



Description des Fichiers PHP :

Voici une description détaillée des principaux fichiers PHP du dossier `pages/` :

`activier_compte.php`

- Active le compte utilisateur via un lien d'activation envoyé par email. Vérifie le token d'activation et met à jour le statut de l'utilisateur dans la base de données.

`admin.php`

- Interface d'administration pour gérer les utilisateurs, les produits et les commandes. Permet à un administrateur de visualiser et modifier les informations stockées dans la base de données.

`ajouter_avis_basket.php`, `ajouter_avis_pantalon.php`,
`ajouter_avis_veste.php`

- Permettent aux utilisateurs d'ajouter des avis et des notes sur les produits spécifiques. Chaque fichier traite les données soumises via un formulaire et insère un nouvel avis dans la base de données.

`ajouter_panier_basket.php`, `ajouter_panier_veste.php`,
`ajouter_panier_pantalon.php`

- Ajoutent les produits respectifs au panier de l'utilisateur. Vérifient les informations du produit et de l'utilisateur avant d'ajouter une entrée dans la table `panier_utilisateur`.

`ajouter_wish_basket.php`, `ajouter_wish_pantalon.php`,
`ajouter_wish_veste.php`

- Ajoutent les produits respectifs à la liste de souhaits de l'utilisateur. Vérifient les informations du produit et de l'utilisateur avant d'ajouter une entrée dans la table `wish_utilisateur`.

`basket.php`, `pantalon.php`, `veste.php`

- Affichent les détails des produits spécifiques. Récupèrent les informations des produits depuis la base de données et génèrent dynamiquement le contenu des pages web.

`cancel.php`

- Gère l'annulation des commandes ou des paiements. Traite les demandes d'annulation et met à jour les informations pertinentes dans la base de données.

`catalogue.php`

- Page d'accueil affichant un catalogue de produits. Récupère une liste de produits depuis la base de données et affiche une vue d'ensemble aux utilisateurs.

`commande.php, order.php, order_detail.php`

- Gèrent les commandes des utilisateurs et affichent les détails des commandes. Ces fichiers traitent les données des commandes, vérifient les informations de paiement, et mettent à jour la base de données en conséquence.

`contact.php, demande_contact.php`

- Fournissent des formulaires de contact pour les utilisateurs. Les demandes soumises sont enregistrées dans la table `demandes_contact` pour traitement ultérieur.

`login.php, logout.php`

- Gèrent l'authentification des utilisateurs. `login.php` traite les informations de connexion soumises et établit une session utilisateur, tandis que `logout.php` détruit la session et déconnecte l'utilisateur.

`panier.php`

- Affiche le panier de l'utilisateur. Récupère les informations des produits dans le panier depuis la table `panier_utilisateur` et génère dynamiquement le contenu de la page.

`process_payment.php, product_payment.php`

- Gèrent les paiements des produits. Vérifient les informations de paiement, communiquent avec l'API de PayPal, et mettent à jour la base de données en conséquence.

`product_catalogue.php, product_pantalon.php, product_basket.php, product_veste.php`

- Affichent les produits dans les différentes catégories. Récupèrent les informations des produits depuis la base de données et génèrent dynamiquement le contenu des pages.

`profil.php`

- Affiche la page de profil de l'utilisateur, incluant les informations personnelles et les points de fidélité. Permet à l'utilisateur de mettre à jour ses informations.

`recupere_code_promo.php`

- Récupère et applique les codes promo. Vérifie la validité des codes promo et applique les réductions correspondantes au panier de l'utilisateur.

`success.php`

- Page affichée après un paiement réussi. Confirme le paiement et affiche les détails de la commande à l'utilisateur.

wish.php

- Affiche la liste de souhaits de l'utilisateur. Récupère les informations des produits dans la liste de souhaits depuis la table `wish_utilisateur` et génère dynamiquement le contenu de la page.

Diagramme UML

Dans le cadre du projet "HB Commerce", nous avons réalisé un diagramme UML pour représenter la structure de notre base de données et les relations entre les différentes entités. Le diagramme UML est un outil essentiel qui nous permet de visualiser et de comprendre les interactions entre les différentes tables de notre base de données, facilitant ainsi la conception et la gestion du système. Il offre une vue d'ensemble claire des dépendances et des connexions, aidant à maintenir une architecture cohérente et efficace.

Vous pouvez consulter le diagramme UML complet dans l'annexe de cette documentation, où un lien vers celui-ci est fourni.


```
sendmail_path = "\"C:\xampp\sendmail\sendmail.exe\" -t"
```

Modifier `sendmail.ini` dans XAMPP:

Localisation: C:\xampp\sendmail\sendmail.ini

```
smtp_server=smtp.gmail.com  
smtp_port=587  
error_logfile=error.log  
debug_logfile=debug.log  
auth_username=ynovmailoff@gmail.com  
auth_password=azkkssxmkrjslbog  
force_sender=ynovmailoff@gmail.com
```

Configuration de PayPal Sandbox :

Pour utiliser le système de paiement, un compte PayPal sandbox doit être utilisé.
Voici un exemple de compte pour les tests :

- Email: sb-ndhzc26594807@personal.example.com
- Mot de passe: 3G@Qd+3K

Fonctionnalités ++ :

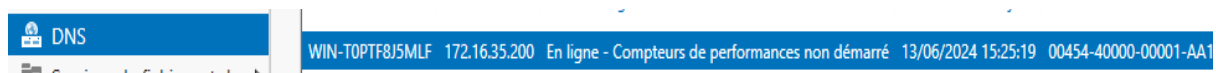
Le projet inclut plusieurs fonctionnalités ++ pour améliorer l'expérience utilisateur :

1. Vérification de compte par mail lors de l'inscription :

- Envoie un email de vérification avec un lien d'activation pour valider l'inscription.
- 2. Barre de recherche sur la page "catalogue.php" :
 - Permet de rechercher des produits par mots-clés partiels (ex. "pa" pour "pantalon").
- 3. Création de cookie à la connexion :
 - Stocke le nom de l'utilisateur dans un cookie pour afficher une popup de bienvenue et un code promo lors de la connexion.
- 4. Système de notation et de commentaires :
 - Permet aux utilisateurs de noter et de commenter les produits dans les pages de détails des produits.
- 5. Code promo opérationnel :
 - Applique des réductions dans la page panier via des codes promo valides.
- 6. Points de fidélité :
 - Les utilisateurs accumulent des points de fidélité qu'ils peuvent échanger contre des récompenses dans la page profil.

Securité (a faire !!)

Tout d'abord nous allons de l'adressage ip que nous avons fait nous avons pris comme DNS le serveur Active directory avec une ip de 172.16.35.200 comme ceci



Ce sera le DNS de toutes les autres machines

Nous avons donc setup la machine Windows serveur (l'AD) comme ceci

Propriétés de : Protocole Internet version 4 (TCP/IPv4)

Général

Les paramètres IP peuvent être déterminés automatiquement si votre réseau le permet. Sinon, vous devez demander les paramètres IP appropriés à votre administrateur réseau.

☐ Obtenir une adresse IP automatiquement

☒ Utiliser l'adresse IP suivante :

Adresse IP : 172 . 16 . 35 . 200

Masque de sous-réseau : 255 . 255 . 255 . 0

Passerelle par défaut : 172 . 16 . 35 . 1

☐ Obtenir les adresses des serveurs DNS automatiquement

☒ Utiliser l'adresse de serveur DNS suivante :

Serveur DNS préféré : 127 . 0 . 0 . 1

Serveur DNS auxiliaire : . . .

☐ Valider les paramètres en quittant

Avancé...

OK Annuler

Avec pour DNS sa propre machine 127.0.0.1 qui est en local.


L'adresse IP 172.16.35.200 pour que les autres machine l'es dans leur paramètres réseau comme DNS

Le masque est en /24

La passerelle fait référence à l'ip de pfsense

Nous allons donc passer a pfsense de suite d'abord le setup de pfsense et ensuite le tableau où l'on a préparer le dhcp ainsi que le firewall

pfsense a étai setup sur deux cartes réseau



Network Adapter	NAT
Network Adapter 2	Host-only

```
Starting syslog...done.
Starting CRON... done.
pfSense 2.7.2-RELEASE amd64 20231206-2010
Bootup complete

FreeBSD/amd64 (pfSense.home.arpa) (ttyv0)

VMware Virtual Machine - Netgate Device ID: cda1365fbb6f94e0ffe

*** Welcome to pfSense 2.7.2-RELEASE (amd64) on pfSense ***

WAN (wan)      -> em0      -> v4/DHCP4: 192.168.45.158/24
LAN (lan)      -> em1      -> v4: 172.16.35.1/24

0) Logout (SSH only)          9) pfTop
1) Assign Interfaces          10) Filter Logs
2) Set interface(s) IP address 11) Restart webConfigurator
3) Reset webConfigurator password 12) PHP shell + pfSense tools
4) Reset to factory defaults    13) Update from console
5) Reboot system               14) Enable Secure Shell (sshd)
6) Halt system                 15) Restore recent configuration
7) Ping host                   16) Restart PHP-FPM
8) Shell

Enter an option: █
```

Le lan donc ce qui représente la passerelle pour les autres machines cela permettra de les relier mais pfsense est tout de même connecter par le WAN donc la une ip donnée directement en par le protocole DHCP.

Maintenant que l'on a la passerelle le DNS nous pouvons créer une machine client sur lequel un utilisateur pourrait utiliser mais aussi pour gérer pfsense directement et pouvoir setup le firewall et le dhcp. Nous avons donc décidé de l'emprunter pour setup pfsense.

Le setup de la machine Windows 10, l'adressage ip :

Propriétés de : Protocole Internet version 4 (TCP/IPv4) ✕

Général

Les paramètres IP peuvent être déterminés automatiquement si votre réseau le permet. Sinon, vous devez demander les paramètres IP appropriés à votre administrateur réseau.

☐ Obtenir une adresse IP automatiquement

☒ Utiliser l'adresse IP suivante :

Adresse IP :

Masque de sous-réseau :

Passerelle par défaut :

☐ Obtenir les adresses des serveurs DNS automatiquement

☒ Utiliser l'adresse de serveur DNS suivante :

Serveur DNS préféré :

Serveur DNS auxiliaire :

☐ Valider les paramètres en quittant Avancé...

OK Annuler

Sur pfsense nous avons fait le dhcp

Primary Address Pool

Subnet

172.16.35.0/24

Subnet Range

172.16.35.1 - 172.16.35.254

Address Pool Range

172.16.35.10

172.16.35.245

From

To

The specified range for this pool must not be within the range configured on any other address pool for this interface.

Additional Pools

+ Add Address Pool

If additional pools of addresses are needed inside of this subnet outside the above range, they may be specified here.

Ainsi que les regles pour L'HTTPS et l'HTTP

pfSense

COMMUNITY EDITION

Firewall / Rules / WAN

Floating

WAN

LAN

Rules (Drag to Change Order)

<input type="checkbox"/>	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input checked="" type="checkbox"/>	0/0 B	*	Reserved Not assigned by IANA	*	*	*	*	*		Block bogon networks	
<input type="checkbox"/>	0/0 B	IPv4 TCP	HTTPS_Access_Machine	*	*	443 (HTTPS)	*	none			

Add

Add

Delete

Toggle

Copy

Save

Separator

pfSense is developed and maintained by Netgate. © ESF 2004 - 2024 View license.

Par la suite nous avons créer la vm pour le serveur DB nous avons utilisé une machine Debian 12 dessus nous avons installer Apache MariaDB phpmyadmin

Tout d'abord il fallait installer le système SSH

```
utilisateurclient@debian:~$ su - root
Mot de passe :
su: Échec de l'authentification
utilisateurclient@debian:~$ su - root
Mot de passe :
root@debian:~# sudo apt install openssh-server
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
openssh-server est déjà la version la plus récente (1:9.2p1-2+deb12u2).
openssh-server passé en « installé manuellement ».
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
root@debian:~# sudo systemctl start ssh
root@debian:~# sudo systemctl enable ssh
Synchronizing state of ssh.service with SysV service script with /lib/systemd
stemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable ssh
root@debian:~#
```

Par la suite nous avons installé le système Apache en trois étapes

```
root@debian:~# sudo apt install build-essential -y
...
root@debian:~# sudo apt install apache2 -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
apache2 est déjà la version la plus récente (2.4.59-1~deb12u1).
apache2 passé en « installé manuellement ».
0 mis à jour, 0 nouvellement installés, 0 à enlever et 0 non mis à jour.
```

Nous avons modifier le fichier pour assurer la sécurité Apache


```

GNU nano 7.2      /etc/apache2/conf-available/security.conf
# ServerTokens
# This directive configures what you return as the Server HTTP response
# Header. The default is 'Full' which sends information about the OS-Type
# and compiled in modules.
# Set to one of:  Full | OS | Minimal | Minor | Major | Prod
# where Full conveys the most information, and Prod the least.
#ServerTokens Minimal
ServerTokens Prod
#ServerTokens Full

#
# Optionally add a line containing the server version and virtual host
# name to server-generated pages (internal error documents, FTP directory
# listings, mod_status and mod_info output etc., but not CGI generated
# documents or custom error documents).
# Set to "EMail" to also include a mailto: link to the ServerAdmin.
# Set to one of:  On | Off | EMail
#ServerSignature Off
ServerSignature Off

```

Nous avons ensuite installer php

```

root@debian:~# sudo apt install php8.2 php8.2-apcu php8.2-bcmath php8.2-bz2 php
8.2-cli php8.2-curl php8.2-gd php8.2-igbinary php8.2-imagick php8.2-intl php8.2-
mbstring php8.2-mysql php8.2-opcache php8.2-pgsql php8.2-readline php8.2-redis p
hp8.2-soap php8.2-tidy php8.2-xml php8.2-xmlrpc php8.2-zip -y

```

MariaDB :

```

root@debian:~# sudo apt install mariadb-server -y

```

```
root@debian:~# sudo mysql
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.11.6-MariaDB-0+deb12u1 Debian 12

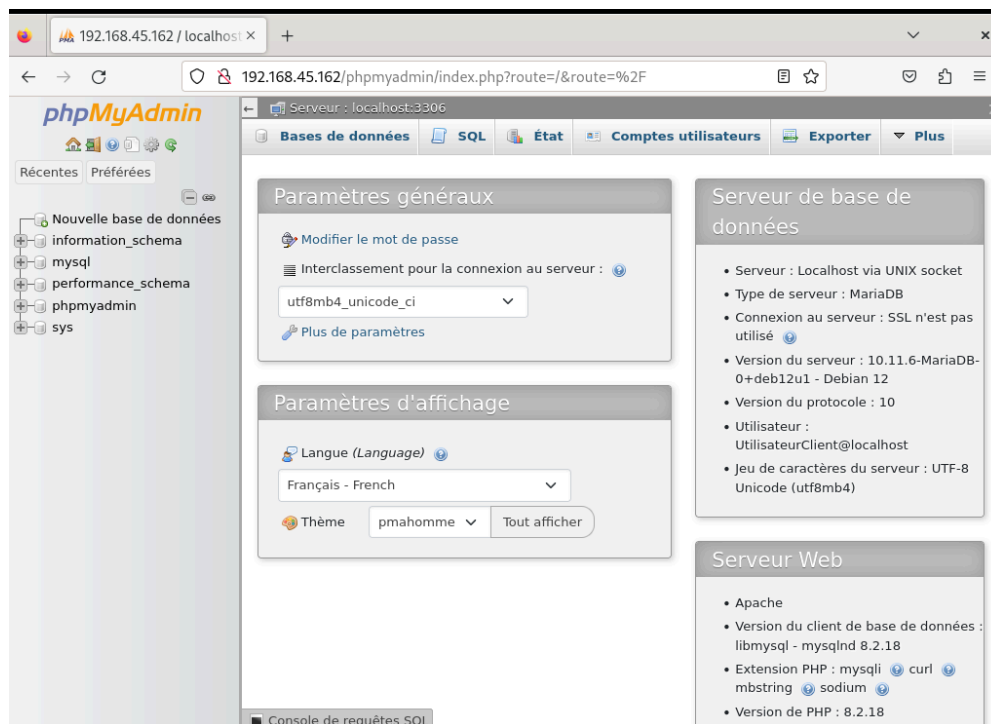
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

et pour finir phpmyadmin :

```
root@debian:~# sudo apt install phpmyadmin -y
```

Et voila l'accès à phpmyadmin et à toutes les db



Conclusion :

Le projet "HB Commerce" est un site e-commerce complet, construit avec une attention particulière à la modularité et à l'organisation. La structure bien définie du

projet, avec ses dossiers distincts pour les fichiers PHP, CSS et les images, permet une gestion facile et une évolution flexible du site. Chaque fichier PHP est dédié à une fonctionnalité spécifique, ce qui simplifie la maintenance et l'ajout de nouvelles fonctionnalités.

La documentation fournie, notamment dans le fichier `readme.md`, offre des instructions claires pour l'installation et la configuration du projet, en détaillant les étapes nécessaires pour configurer XAMPP, le système de paiement PayPal sandbox et les divers paramètres de configuration. Cette documentation est essentielle pour garantir que le projet peut être rapidement mis en place et utilisé efficacement.

En outre, la base de données est soigneusement structurée, avec des tables bien définies et des relations claires entre elles. Cela assure une gestion efficace des utilisateurs, des produits, des commandes, et des avis, tout en permettant des fonctionnalités avancées comme les codes promo, les points de fidélité, et les échanges de récompenses.

En somme, "HB Commerce" est un projet bien pensé et bien exécuté, offrant une solution e-commerce robuste et flexible. La documentation technique détaillée et la structure organisée du projet facilitent non seulement l'utilisation actuelle mais aussi les futures extensions et améliorations, assurant ainsi la pérennité et le succès du site.

Ce projet marque une étape importante dans l'amélioration de notre infrastructure IT. En combinant des technologies robustes comme Active Directory, pfSense (incluant les fonctionnalités DHCP), un serveur DNS, et un serveur web avec Apache2, MariaDB, et phpMyAdmin, nous avons pu créer un espace client sécurisé, performant et facile à gérer. Nous sommes confiants que cette mise en place répondra aux besoins de nos clients tout en assurant une protection maximale de leurs données.

Annexe :

GitHub : [baayvin17/Projet_Final_Commerce \(github.com\)](https://github.com/baayvin17/Projet_Final_Commerce)

Trello:

<https://trello.com/invite/b/gxvEpnmb/ATTI07b55c5f8cc086bbfa3634eab74433015E9F9F2F/projetfinalcommerce>

Uml : [UML_Projet_Final_Commerce.drawio - draw.io \(diagrams.net\)](https://draw.io)